Polecenie: Konwersja liczb arabskich na rzymskie (i z powrotem). Napisać program
konwertujący liczbę arabską (czytaną z klawiatury) na rzymską, oraz podobny działający w odwrotnym kierunku.

Opis programu podzielę na zamianę z systemu rzymskiego na arabski oraz z arabskiego na rzymski.
Ale na starcie zdefiniowałem biblioteki, podstawową <stdio.h>, <stdlib>, <string> z której będę używać funkcje w dalszej części programu oraz biblioteki <ctype.h> która była podana w poleceniu do zadania i biblioteka matematyczna <math.h>.

Na start w programie jest zawarta struktura danych która była zawarta w poleceniu:

```
struct RZYM {
int arab;
char *rzym;
} rz[]={{1,"I"},{4,"IV"},{5,"V"},{9,"IX"},{10,"X"},
{40,"XL"},{50,"L"},{90,"XC"},{100,"C"},
{400,"CD"},{500,"D"},{900,"CM"},{1000,"M"}};
```

ZAMIANA Z RZYMSKIEGO NA ARABSKI:

Na początku zdefiniowałem funkcję `int valueOfRome(char value)`

która kolokwialnie mówiąc „jedzie” po strukturze rz[i] od i=12 czyli od jej końca i sprawdza czy wartość będzie różna od wprowadzonych danych.

Następnie w programie jest zawarta kolejna funkcja w której ciele znajduje się instrukcja warunkowa switch. Nie będę tutaj opisywał każdej linijki po kolei ponieważ sam switch jest dosyć rozbudowany, ale przedstawie koncept działania.

Całość sprawdza przy liczbach I,X oraz C czy występuje przed nimi wartość z którą może się połączyć. Czyli przykładowo dla X program sprawdza czy występuje przed nim L albo C. Jeśli tak się stało to wtedy musi te liczby ze sobą połączyć tak żeby z XL powstało nam 40. Natomiast jeśli tak się nie dzieje to funkcja to pomija i przechodzi do sprawdzenia następnego znaku i tak do końca. Co do wartości z którymi liczby I,X oraz C mogą się łączyć są to liczy:

I może połączyć się z: V lub X
X może połączyć się z: L lub C
C może połączyć się z: D lub M

Więc podsumowując jeśli program znajduje takie połączenia to wtedy dodaje ich wartość do sumy, a samą wartość pobiera z wcześniej zainicjowanej struktury. Jeśli nie ma połączenia to program doda wartość normalną (czyli bez połączenia) do sumy którą pobierze ze struktury. Ważne jest to że jeśli dojdzie do połączenie program przeskakuje o 2 liczby tak aby drugi raz nie przeliczyć wartości z połączenia.

ZAMIANA Z ARABSKIEGO NA RZYMSKI:

Na początku zdefiniowałem funkcję `char *arabToRome(int n)`
Gdzie dynamicznie zalokowałem pamięć dla `romeNumbers = (char *)malloc(sizeof(char)*10)`
Po to aby później móc wpisywać do niej wartości ze struktury. Następnie określiłem dużą pętle while od n w której wpisałem dwie mniejsze pętle while. Całość działa na zasadzie że jeśli pobrana wartość z systemu arabskiego jest mniejsza od danego wiersza struktury to zmienna k się zwiększa. A następna pętla działa na zasadzie podobnej lecz pobiera dane z wcześniejszej pętli. Funkcja na końcu zwraca wartość w systemie rzymskim.

Następnie określiłem funkcję główną main() która pobiera dane od użytkowania i wprowadza je do tablicy char'ów oraz program sprawdza ich długość. Dalej dzięki zmiennej int isArab i użyciu funkcji `isdigit` sprawdzam jaka jest pierwsza wartość tej tablicy, jeśli jest to cyfra to program zwróci liczbę różną od zera, w przeciwnym wypadku zwraca 0. Dzięki temu wiem z czego program powinien zamienić na jaki system liczbowy bez konieczności osobnego wpisywania tej informacji.

A więc jeśli jest to liczba z systemu arabskiego na rzymski to muszę przerzucić tablicę char z daną liczbą na wartość int w tym celu używam pętli while:

```
for (int h = 0; h < lenghOfInput; h++) //h to licznik po tablicy char
{
liczba = liczba + (numberInput[h] - 48) * pow(10, lenghOfInput - 1 - h);
}
```

Która w prosty sposób przelicza każdą liczbę po kolei na int i mnoży ją o wartość 10 do potęgi na której dana liczba stoi w tabeli. Dzięki temu program posiada daną liczbe w zmiennej int.

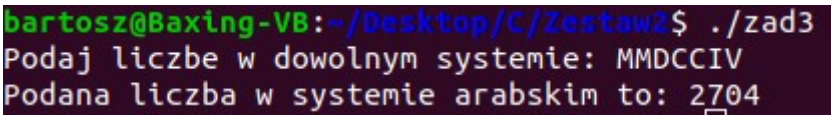
Następne ważne jest żeby zakres liczb ograniczyć od 0 do 3999 ponieważ tylko da tylu działa ten program.

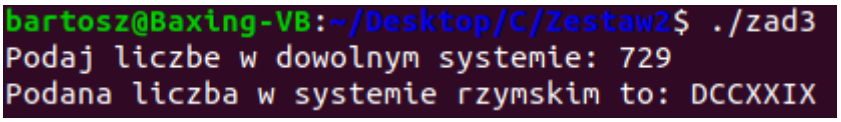
W zamianie z liczb rzymskich na arabskie wystarczy podać wartości funkcji już bez konwertowania tablicy char na 1 zmienną int.

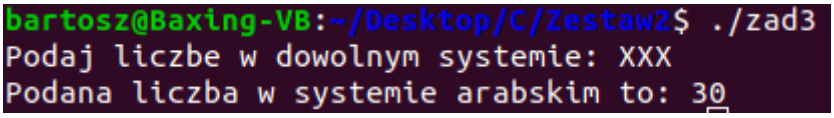
PRZYKŁAD DZIAŁANIA PROGRAMU:

(UWAGA, POPRZEZ ZASTOSOWANIE FUNKCJI MATEMATYCZNEJ pow(x,x) przy kompilacji należy użyć dodatkowo -lm)

```
bartosz@Baxing-VB:~/Desktop/C/Zestaw2$ gcc zad3.c -o zad3 -lm
bartosz@Baxing-VB:~/Desktop/C/Zestaw2$ ./zad3
Podaj liczbe w dowolnym systemie: 1886
Podana liczba w systemie rzymskim to: MDCCCLXXXVI
```







Jak widać na powyższych przykładach program sam wie w jakim systemie dostaje liczbę i na jaką musi ja zamienić.